Project no.: **ICT-FP7-STREP-214755**

Project full title: **Quantitative System Properties in Model-Driven Design**

Project Acronym: **QUASIMODO**

Deliverable no.: D2.4

Title of Deliverable: Abstraction and Refinement

Contractual Date of Delivery to the CEC:	**Month 30**
Actual Date of Delivery to the CEC:	**Month 40 (June 1, 2011)**
Organisation name of lead contractor for this deliverable:	**P02 ESI/RU**
Author(s):	**Frits Vaandrager, Holger Hermanns, Joost-Pieter Katoen, Kim Larsen, Mariëlle Stoelinga, and Mark Timmer.**
Participant(s):	**P01 AAU, P02 ESI/RU, ESI/UT, P04 RWTH, P05 SU**
Work package contributing to the deliverable:	**WP2**
Nature:	**R**
Version:	**1.0**
Total number of pages:	**30**
Start date of project:	1 Jan. 2008 **Duration:** 36 month

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)

Dissemination Level

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract:

This deliverable presents an overview of the work on abstraction and refinement that has been carried out within Task 2.2 during Year 3 of the QUASIMODO project.

Keyword list: Model Reduction, Refinement, Abstraction, Compositionality, Counterexample-guided abstraction refinement, Probabilistic Automata, Timed Automata, Hybrid Automata, Continuous Time Markov chains, Interactive Markov chains, Markov Automata

Contents

Abbreviations

AAU: Aalborg University, DK

ESI: Embedded Systems Institute, NL

ESI/RU: Radboud University Nijmegen, NL

ESI/UT: University of Twente, NL

RWTH: RWTH Aachen University, D

SU: Saarland University, D

1 Introduction

This deliverable presents an overview of the work on abstraction and refinement that has been carried out within Task 2.2 during Year 3 of the QUASIMODO project. Before we turn to the abstraction and refinement, we first give a very high-level overview of the modelling formalisms studied within the project.

1.1 Quantitive models for reactive systems

QUASIMODO has studied frameworks for modeling and analysis of *dynamic behavior* of *reactive systems*. These reactive systems are described in terms of *states* and *transitions* between states. Roughly speaking, four different types of transitions are being considered.

Discrete transitions (D) Within traditional theories of concurrency, such as Milner's CCS, systems may perform *discrete transitions* of the form

$$s \xrightarrow{a} s'$$

where s and s' are states and a is an action label. Process calculi study the resulting *labelled transition systems (LTSs)*, and define operators to combine these labeled transition systems, such as a parallel composition operator in which a transitions from different component LTSs may synchronize.

Probabilistic transitions (P) The consideration of stochastic phenomena has led to the development of a plethora of stochastic and probabilistic process calculi. One prominent calculus is that of *probabilistic automata (PA)* [80], which extends classical concurrency models in a simple yet conservative fashion, and comes equipped with a compositional theory for strong and weak bisimulation, and corresponding equational axiomatizations. In probabilistic automata (*PA*), concurrent processes may perform random experiments inside a transition. This is represented by transitions of the form

$$s \xrightarrow{a} \mu$$

where s is a state, a is an action label, and μ is a probability distribution on states. Labelled transition systems are instances of this model family, obtained by restricting to Dirac distributions (assigning full probability to single states). Thus, foundational concepts and results of standard concurrency theory are retained in their full beauty, and extend smoothly to the model of probabilistic automata.

Since the model is akin to Markov decision processes, its fundamental beauty can be paired with powerful model checking techniques, as implemented for instance in the PRISM tool. *Discrete time Markov chains (DTMC)* are obtained as a special case of PAs in which, for each state s, there is just a single outgoing probabilistic transition.

Random Delays (RD) *Interactive Markov chains (IMC)* in turn arise from classical concurrency models by allowing, besides discrete transitions, a second type of transitions

$$s \xrightarrow{\lambda} s'$$

that can embody random delays governed by a negative exponential distribution with some parameter λ. This twists the model to one that is running on a continuous time line, and where executions of actions take no time and happens immediately – unless an action can be blocked by the environment. This is linked to the process algebraic notion of maximal progress for internal actions. By dropping the second type of transitions, again, standard concurrency theory is regained in its entirety, and extends smoothly to the full *IMC* model. The availability of tool support has led to several academic and industrial applications, see e.g. [79].

Trajectories (H/T) *Hybrid automata (HA)* models arise from classical concurrency models by introducing, besides discrete transitions, *trajectories*. A trajectory is a function

$$\tau : I \to S$$

from some left-closed interval I over the real numbers to the set of states S. The idea is that discrete transitions occur instantaneously, whereas trajectories describe the the (continuous) evolution of the system over time. Alur-Dill style *timed automata (TA)* constitute a special case of hybrid automata in which only a very restricted type of trajectories are allowed: states are valuations of *discrete variables* (which remain unchanged along a trajectory) and *clock variables* (which always increase with rate 1). In an Alur-Dill automaton, trajectories are fully determined by their initial state, their length, and whether they are right-open or right-closed.

Within Quasimodo, we studied abstraction and refinement for the quantitative extensions of labeled transition systems described above (DTMCs, CTMCs, PAs/MDPs, IMCs, HAs, TAs), but also for new combinations of the basic transition types such as **P+RD** (the model of Markov Automata (MA) described in Section 2.4), and **P+H** (the model of Probabilistic Hybrid Systems (PHS) described in Section 3.8.

We should apologize for some terminological confusion created by the authors of this deliverable. Firstly, the acronym IMC is used both for Interactive Markov Chains (for instance in Section 3.7) and for Interval Markov Chains (Section 3.1). Secondly, the term Timed I/O Automata is used both for for an extension of Alur-Dill timed automata with I/O distinction that has been implemented in the ECDAR tool and is described in Sections 3.2 and 3.3, and for the very expressive (hybrid) model of Lynch et al proposed in Section 3.4.[1]

[1]The model of Lynch et al is much more expressive than the model behind the ECDAR tool. But whereas refinement checking is fully automatic in ECDAR, tool support for the model of Lynch et al through the Tempo toolset is limited. Unlike the model of Lynch et al, ECDAR does not assume that input actions are always enabled. This makes ECDAR an appropriate tool for the specification of timed interfaces. Even though the two modelling frameworks focus on different aspects, they are fully compatible and we expect that it will be possible to integrate the two frameworks into a unified theory of timed I/O automata.

1.2 Abstraction techniques

In the modeling and analysis of systems, abstraction is inherently of great importance. The act of modelling is itself already an act of abstraction, in the sense that only relevant information of the object of modeling finds entry into the model. Frequently, abstraction is an indispensible means to make the analysis of systems feasible, usually because the state space of the system model to be analysed is either too large, or even infinite. But abstraction — the word basically is synonymous with "throwing information away" — may also introduce imprecision in the results obtainable from the analysis of the abstract system: they may be inconclusive or plain wrong.

Model reduction and bisimulations Before starting to throw away information (and running the risk to abstract too much), it makes sense to first exploit the fact that often different states in a model are behaviorally equivalent / bisimilar. Bisimulation equivalences can be a very powerful tool to reduce the space of systems, especially when applied compositionally. Notions of bisimulation have been proposed before for the various model classes that have been studied within QUASIMODO, except for the new model class of Markov automata. Section 2.4 proposes a compositional, weak notion of bisimulation for Markov automata. In addition, the project has made significant progress in automatic bisimulation reductions for probabistic systems with data (Sections 2.2 and 2.3), and in linking bisimulations and compositional proof systems for a general class of continuous-time and continuous-space Markov processes (Section 2.1).

Abstraction, Refinement and Compositionality In order to make sure that certain properties carry over from an abstract model to a concrete model, we need to establish a formal relationship between these models. For this usually the concept of abstract interpretation is used, or notions of (weak, alternating) simulation. An advantage of simulation relations is that, usually, they are compositional, and thus support a stepwise refinement approach. In Section 3 of this deliverable, we report on a series of results on new (compositional) abstraction techniques for different model classes.

Counterexamples and CEGAR Within the computer-aided verification community, a popular technique to deal with huge, symbolic state spaces is *counterexample guided abstraction-refinement (CEGAR)* [64, 18, 19]. In the CEGAR approach, an initial very course abstraction of a system is computed automatically. In this abstraction, for instance, the only information about an integer variable that is preserved is whether it is zero, positive or negative. Or, alternatively, all valuations of program variables that cannot be distinguished by any of the Boolean guards that occur in a program are deemed equivalent. Next exhaustive state space search (model checking) is used to explore the abstract model. If in the abstract model no "bad" state can be reached then we know by construction that no bad state can be reached in the original model. In this case we have established correctness of the model, and we are done. In case a bad state can be reached in the abstract model there are two possibilities:

1. either there is a corresponding execution of the original model that leads to a bad state; this means that we have found a bug in the original model,

	Bisimulation	Refinement	CEGAR
PA	2.2, 2.3	3.1	4.2
DTMC		3.5	4.1
CTMC	2.1	3.5, 3.6	
IMC		3.7	
MA	2.4		
TA		3.2, 3.3	
HA		3.4	
PHS		3.8	

Figure 1: Overview of results; numbers refer to subsections

2. or the bad execution in the abstract model does not correspond to any execution in the original model; in this case we can use the information about the failed correspondence to construct a refinement of the abstraction, that is, a new abstraction that is in between the old abstraction and the original model, and we repeat the analysis.

CEGAR based software model checkers such as SLAM and Blast have been succesfully applied within the domain of debugging of device drivers (programs with over 100,000 lines of C code). CEGAR techniques has also been developed for the (computationally difficult) analysis of hybrid systems [17]. A prerequisite for applying CEGAR techniques is the availability of counterexamples. In the case of probabilistic systems, it is not at all obvious what constitutes a good counterexample. A major contribution of QUASIMODO, therefore, is the definition of a notion of counterexample for probabilistic systems based on the idea of strongest evidence (Section 4.1). Several groups have used these results to develop CEGAR algorithms for Markov decision processes [16, 60]. Within QUASIMODO, we have used these ideas to develop a CEGAR based analysis tool for potentially infinite Markov decision processes (Section 4.2). It is an important topic for future research to extend these results to other/richer probabilistic models.

Overview of results Figure 1 presents an overview of the results reported in this deliverable. Numbers refer to the subsections in which the results are described in more detail.

2 Model Reductions and Bisimulation

2.1 Compositional proof system for Markovian models

Participants

- Radu Mardare, Kim G. Larsen; AAU
- Luca Cardelli; Microsoft Research Cambridge, UK

Contribution Complex networks (e.g., embedded systems, communication networks, the Internet etc.) and complex systems (e.g., biological, ecological, social, financial, etc.) are often modeled as stochastic processes, to encapsulate a lack of knowledge or inherent randomness. Such systems are frequently modular in nature, consisting of parts which are systems in their own right. Their global behaviour depends on the behaviour of their parts and on the links which connect them. Understanding such systems requires integration of local stochastic information in a formal way, in order to address questions such as: to what extent is it possible to derive global properties of the system from the local properties of its modules?.

In [72], we introduce Modular Markovian Logic (MML) for compositional continuous-time and continuous-space Markov processes. MML combines operators specific to stochastic logics with operators that reflect the modular structure of the semantics, similar to those used by spatial and separation logics. We present a complete Hilbert-style axiomatization for MML, prove the small model property and analyze the relation between the stochastic bisimulation and the logical equivalence relation induced by MML on models.

2.2 A linear process-algebraic format with data for probabilistic automata

Participants

- Joost-Pieter Katoen; RWTH
- Jaco van de Pol, Mariëlle Stoelinga, Mark Timmer; ESI/UT

Challenge Efficient model checking algorithms exist, supported by powerful software tools, for verifying qualitative and quantitative properties for a wide range of probabilistic models. While these techniques are important for areas like security, randomised distributed algorithms, systems biology, and dependability and performance analysis, two major deficiencies exist: the *state space explosion* and the restricted treatment of *data*.

Unlike process calculi like μCRL [38] and LOTOS NT [34], which support rich data types, modelling formalisms for probabilistic systems mostly treat data as a second-class citizen. Instead, the focus has been on understanding random phenomena and the interplay between randomness and nondeterminism. Data is treated in a restricted manner: probabilistic process algebras typically only allow a random choice over a fixed distribution, and input languages for probabilistic model checkers such as the reactive module language of PRISM [48] or the probabilistic variant of Promela [1] only support basic data types, but neither support more advanced data structures. To model realistic systems, however, convenient means for data modelling are indispensable.

Additionally, although parameterised probabilistic choice is semantically well-defined [11], the incorporation of data yields a significant increase of, or even an infinite, state space. However, current probabilistic minimisation techniques are not well-suited to be applied in the presence of data: aggressive abstraction techniques for probabilistic models (e.g., [20, 25, 46, 52, 66, 59]) reduce at the model level, but the successful analysis of data requires *symbolic* reduction techniques. Such methods reduce stochastic models using syntactic transformations at the *language*

level, minimising state spaces *prior to* their generation while preserving functional and quantitative properties. Other approaches that partially deal with data are probabilistic CEGAR [47, 58] and the probabilistic GCL [73].

Our aim was to develop symbolic minimisation techniques — operating at the syntax level — for data-dependent probabilistic systems.

Results In [53, 54, 55], we defined a probabilistic variant of the process-algebraic μCRL language [38], named prCRL, which treats data as a first-class citizen. The language prCRL contains a carefully chosen minimal set of basic operators, on top of which syntactic sugar can be defined easily, and allows data-dependent probabilistic branching. Because of its process-algebraic nature, message passing can be used to define systems in a more modular manner than with for instance the PRISM language.

To enable symbolic reductions, we provided a two-phase algorithm to transform prCRL terms into LPPEs: a probabilistic variant of *linear process equations* (LPEs) [6], which is a restricted form of process equations akin to the Greibach normal form for string grammars. We proved that our transformation is correct, in the sense that it preserves strong probabilistic bisimulation [67]. Similar linearisations have been provided for plain μCRL [12], as well as a real-time variant [83] and a hybrid variant [85] therefore.

To motivate the advantage of the LPPE format, we draw an analogy with the purely functional case. There, LPEs have provided a uniform and simple format for a process algebra with data. As a consequence of this simplicity, the LPE format was essential for theory development and tool construction. It led to elegant proof methods, like the use of invariants for process algebra [6], and the cones and foci method for proof checking process equivalence [39, 33]. It also enabled the application of model checking techniques to process algebra, such as optimisations from static analysis [36] (including dead variable reduction [84]), data abstraction [32], distributed model checking [8], symbolic model checking (either with BDDs [10] or by constructing the product of an LPE and a parameterised μ-calculus formula [37, 40]), and confluence reduction [9] (a variant of partial-order reduction). In all these cases, the LPE format enabled a smooth theoretical development with rigorous correctness proofs (often checked in PVS), and a unifying tool implementation. It also allowed the cross-fertilisation of the various techniques by composing them as LPE to LPE transformations.

We already generalised several reduction techniques from LPEs to LPPEs: constant elimination, summation elimination, expression simplification, dead variable reduction, and confluence reduction. The generalisation of these techniques turned out to be very elegant. Also, we implemented SCOOP: a tool that can linearise prCRL models to LPPE, automatically apply all these reduction techniques, and generate state spaces. Experimental validation, using several variations of two benchmark protocols for probabilistic model checking, show that state space reductions of up to 95% can be achieved.

Perspective Our prCRL language and the LPPE format can be seen as the first essential step towards the symbolic minimisation of probabilistic state spaces, as well as the analysis of parameterised probabilistic protocols. Our results show that the treatment of probabilities is simple

and elegant, and rather orthogonal to the traditional setting [83] (which is very desirable, as it simplifies the generalisation of existing techniques to the probabilistic setting).

The LPPE format already led to the generalisation of dead variable reduction and confluence reduction to the probabilistic setting, and we demonstrated by a case study remarkable results in reducing both a system's state space and the time needed to generate it.

Interesting directions for future work are the development of additional minimisation techniques, the application of proof techniques such as the cones and foci method to LPPEs, and the investigation of abstraction methods in the context of LPPEs. Also, more case studies could be conducted, to evaluate the effects of our reduction techniques.

2.3 Confluence reduction for probabilistic systems

Participants

- Mark Timmer, Mariëlle Stoelinga, Jaco van de Pol; ESI/UT

Challenge Model checking of probabilistic systems is getting more and more attention, but there still is a large gap between the number of techniques supporting traditional model checking and those supporting probabilistic model checking. Especially methods aimed at reducing state spaces are greatly needed to battle the omnipresent state space explosion.

In this work, we generalise the notion of confluence [41] from labelled transition systems (LTSs) to probabilistic automata (PAs) [80]. Basically, we define under which conditions unobservable transitions (often called τ-transitions) do not influence a PA's behaviour (i.e., they commute with all other transitions). Using this new notion of probabilistic confluence, we are able to introduce a symbolic technique that reduces PAs while preserving branching probabilistic bisimulation.

The non-probabilistic case. Our methodology follows the approach for LTSs from [9]. It consists of the following steps: (i) a system is specified as the parallel composition of several processes with data; (ii) the specification is linearised to a canonical form that facilitates symbolic manipulations; (iii) first-order logic formulas are generated to check symbolically which τ-transitions are confluent; (iv) an LTS is generated in such a way that confluent τ-transitions are given priority, leading to an on-the-fly (potentially exponential) state space reduction. Refinements by [75] make it even possible to perform confluence detection on-the-fly by means of boolean equation systems.

Results In [81, 82] we introduced three novel notions of probabilistic confluence. Inspired by [7], these are *weak probabilistic confluence*, *probabilistic confluence* and *strong probabilistic confluence* (in decreasing order of reduction power, but in increasing order of detection efficiency).

We proved that the stronger notions imply the weaker ones, and that τ-transitions that are confluent according to any of these notions always connect branching probabilistically bisimilar states. Basically, this means that they can be given priority without losing any behaviour. Based

on this idea, we proposed a reduction technique that can be applied using the two stronger notions of confluence. For each set of states that can reach each other by traversing only confluent transitions, it chooses a representative state that has all relevant behaviour. We proved that this reduction technique yields a branching probabilistically bisimilar PA. Therefore, it preserves virtually all interesting temporal properties.

As we want to analyse systems that would normally be too large, we need to detect confluence symbolically and use it to reduce on-the-fly during state space generation. That way, the unreduced PA never needs to be generated. Since it is not clear how not to detect (weak) probabilistic confluence efficiently, we only provided a detection method for strong probabilistic confluence. Here, we exploit a previously defined probabilistic process-algebraic linear format, which is capable of modelling any system consisting of parallel components with data [53]. In this paper, we show how symbolic τ-transitions can be proven confluent by solving formulas in first-order logic over this format. As a result, confluence can be detected symbolically, and the reduced PA can be generated on-the-fly. We presented a case study of leader election protocols, showing significant reductions.

Related work. As mentioned before, we basically generalise the techniques presented in [9] to PAs.

In the probabilistic setting, several reduction techniques similar to ours exist. Most of these are generalisations of the well-known concept of partial-order reduction (POR) [76]. In [3] and [21], the concept of POR was lifted to Markov decision processes, providing reductions that preserve quantitative LTL$\setminus$X. This was refined in [2] to probabilistic CTL, a branching logic. Recently, a revision of POR for distributed schedulers was introduced and implemented in PRISM [35].

Our confluence reduction differs from these techniques on several accounts. First, POR is applicable on state-based systems, whereas our confluence reduction is the first technique that can be used for action-based systems. As the transformation between action- and state-based blows up the state space [74], having confluence reduction really provides new possibilities. Second, the definition of confluence is quite elegant, and (strong) confluence seems to be of a more local nature (which makes the correctness proofs easier). Third, the detection of POR requires language-specific heuristics, whereas confluence reduction acts at a more semantic level and can be implemented by a generic theorem prover. (Alternatively, decision procedures for a fixed set of data types could be devised.)

Our case studies showed that the reductions obtained using probabilistic confluence exceed the reductions obtained by probabilistic POR [42].

Perspective We introduced three new notions of confluence for probabilistic automata. We first established several facts about these notions, most importantly that they identify branching probabilistically bisimilar states. Then, we showed how probabilistic confluence can be used for state space reduction. As we used representatives in terminal strongly connected components, these reductions can even be applied to systems containing τ-loops. We defined how confluence can be detected in the context of a probabilistic process algebra with data by proving formulas in first-order logic. This way, we enabled on-the-fly reductions when generating the state space

corresponding to a process-algebraic specification. A case study illustrated the power of our methods.

2.4 Markov automata

This work is published in *LICS 2010* [31], and extended in *CONCUR 2010* [30].

Participants

- Christian Eisentraut, Holger Hermanns; SU
- Lijun Zhang, DTU Copenhagen

Challenge The seemingly simple question addressed in this work is: What happens if we integrate the theories of Interactive Markov Chains (IMC) and Probabilistic Automata (PA)? This is not only an academic question, since industrial engineers are desperately asking for formalisms that support both, probabilistic branching and exponentially distributed delays. Therefore, we are looking into a model class *MA* (Markov automata), that supports both the probabilistic transitions $s \xrightarrow{a} \mu$ of PAs, and the random delay transitions of $\xrightarrow{\lambda} s'$ of IMCs.

In the context of Petri nets, this move has been done 25 years ago. After Molloy introduced Stochastic Petri nets (which correspond to continuous time Markov chains), it was a matter of two years until also probabilistic branching was supported in the form of weighted immediate transitions, leading to the model of generalised stochastic Petri nets (GSPNs). However, the inventors of GSPNs initially overlooked the issue of non-determinism arising from concurrently enabled probabilistic branching. To date, the analysis trajectory for GSPNs is a partial one. It is restricted to confusion-free nets, a class of nets, where non-determinism is absent. Still, the analysis trajectory developed for this class gives us important inspiration when formulating our theory.

Results While a direct combination of the *PA* and the *IMC* theories is an almost easy exercise, it turns out to be very demanding if reflecting on the different time scales we now work in. As in plain *IMC*s, internal probabilistic transitions cannot be blocked and take no time to execute. Consequently, we aim at fusing sequences of them. This implies that we need to partially ignore the branching structure of our probabilistic automata induced substructures when defining equalities, especially weak bisimulation, on them.

The main contribution of this work is a definition of weak bisimulation on the *MA* model that is (i) (indeed) an equivalence satisfying a number of desirable equalities, (ii) a congruence with respect to parallel composition, (iii) a conservative extension of *IMC* weak bisimulation, (iv) coarser than *PA* weak bisimulation, and (v) can serve as a correctness criterion when associating a continuous time Markov chain to a confusion free GSPN.

The interplay of random phenomena and continuous dynamics deserves increased attention especially in the context of wireless sensing and control applications. The analysis of properties of such systems thus needs to take into account probabilistic variations of systems with hybrid

dynamics. In safety verification of classical hybrid systems one is interested in whether a certain set of unsafe system states can be reached from a set of initial states. In the probabilistic setting, we may ask instead whether the probability of reaching unsafe states is below some given threshold. Other important properties relate to the performance of systems, such as the average data transmission rate of a wireless multimedia application on the long run, or the expected maximal time a wireless sensing network needs to transmit a measurement to a base station.

Perspective The Markov automata model appears as a very important step in the quest for compositional and expressive performance models. The research agenda related to this model class is wide open. Particularly interesting are questions related to decidability and decision algorithms for weak bisimulation, as well as studies relating the *MA* model to Markov decision processes in continuous time. The latter might pave the way for a complete analysis trajectory for *MA* that is as yet not available.

3 Abstraction, Refinement and Compositionality

3.1 Probabilistic systems

Participants

- Benoit Caillaud, Benoit Delahay, Axel Legay; INRIA/IRISA, France
- Kim G. Larsen, Mikkel Larsen Pedersen; AAU
- Andrzej Wasowski; IT University, Copenhagen, Denmark
- Joost-Pieter Katoen, Falek Sher; RWTH

Contribution In the early work [51] the formalism of Interval Markov chains was introduced as an extension of Modal Transition Systems to the setting of probabilistic systems, allowing for notions of satisfaction and refinement to generalize well-established notions of probabilistic bisimulation. Informally, IMCs extend Markov Chains by labeling transitions with intervals of allowed probabilities rather than concrete probability values. In [29] we study complexity of several problems for IMCs. In particular we close the complexity gap for thorough refinement of two IMCs and for deciding the existence of a common implementation for an unbounded number of IMCs, showing that these problems are EXPTIME-complete. We also prove that deciding consistency of an IMC is polynomial and discuss suitable notions of determinism for such specifications.

However, the expressive power of IMCs is inadequate as it supports neither logical nor structural composition. During the second year of Quasimodo we introduced in [13] the notion of *Constraint Markov Chains* (CMCs) as a foundation for component-based design of probabilistic systems. We provided constructs for refinement, consistency checking, logical and structural composition of CMC specifications – all indispensable ingredients of a compositional design

methodology. During the third year of Quasimodo a full version of [13] has been accepted for publication in Theoretical Computer Science [14].

In [26, 27] we propose the formalism of Abstract Probabilistic Automata (APA) that may be viewed as a combination of Modal Transition Systems and Constraint Markov Chains providing abstraction for both transition systems and Markov Chains. In APAs uncertainty of the non-deterministic choices is modeled by may/must modalities on transitions while uncertainty of the stochastic behaviour is expressed by (underspecified) stochastic constraints. We have developed a complete abstraction theory for PAs, and also propose the first specification theory for them. Our theory supports both satisfaction and refinement operators, together with classical stepwise design operators. In addition, we study the link between specification theories and abstraction in avoiding the state-space explosion problem.

The theory of APA is equipped with a series of aggressive abstraction techniques for state-space reduction as well as a specification theory for both logical and structural comparisons. In [28], we present the tool APAC for reasoning about Abstract Probabilistic Automata.

3.2 ECDAR-style timed I/O automata

Participants

- Alexandre David, Kim G. Larsen, Ulrik Nyman; AAU
- Axel Legay; INRIA/IRISA, France
- Andrzej Wasowski; IT University, Copenhagen, Denmark

Contribution Many modern systems are big and complex assemblies of numerous components. The components are often designed by independent teams, working under a common agreement on what the interface of each component should be. Consequently, compositional reasoning, a mathematical foundations of reasoning about interfaces, is an active research area. Specification theories should support various features including (1) refinement, which allows to compare specifications as well as to replace a specification by another one in a larger design, (2) logical conjunction expressing the intersection of the set of requirements expressed by two or more specifications, (3) structural composition, which allows to combine specifications, and (4) last but not least, a quotient operator that is dual to structural composition. The latter is crucial to perform incremental design.

In [23], we developed a complete specification framework for real-time systems using Timed I/O Automata as the specification formalism, with the semantics expressed in terms of Timed I/O Transition Systems. In [22], the tool ECDAR offers an implementation of the theory on top of the engine for timed games, Uppaal-Tiga, supporting the operations of composition, conjunction, and refinement.

In [24], we propose a new efficient algorithm for checking Büchi objectives of timed games. We show that this new algorithm can be used to strengthen the infinite behavior of an interface, or to guarantee that the interface can indeed be implemented. Our theory has been implemented in the ECDAR tool. The contributions are:

1. A new on-the-fly algorithm for checking Büchi objectives of two-player timed games. The algorithm builds on an existing efficient algorithm for solving reachability objectives [15, 5], but it uses zones as a symbolic representation. We show how the algorithm can be combined with a safety objective. This allows, for example, to guarantee that a player has a strategy to stay within a set of states without blocking the progress of time. Similar results were proposed by de Alfaro et al. but for a restricted class of timed interfaces and without an implementation of the continuous case.

2. A realistic case study. Most existing interface theories have not been implemented and evaluated on concrete applications. We use ECDAR to show that our interface theory indeed is a feasible solution for the design of potentially complex timed systems. More precisely, we specify an infrared sensor for measuring short distances and for detecting obstructions. The extensive case study reveals both the advantages and disadvantages of our theory.

3.3 Weighted systems

Participants

- Line Juhl, Kim G. Larsen; AAU
- Sebastian Bauer; Ludwig-Maximilians-Universitat, München, Germany
- Uli Fahrenberg, Axel Legay; INRIA/IRISA, France.

Contribution Modal transition systems are labeled transition systems equipped with two types of transitions: must transitions which are mandatory for any implementation, and may transitions which are optional for implementations. It is well admitted that modal transition systems match all the requirements of a reasonable specification theory (see e.g. [78] for motivations). Also, practical experience shows that the formalism is expressive enough to handle complex industrial problems.

In [4], we introduce a novel formalism of label-structured modal transition systems that combines the classical may/must modalities on transitions with structured labels that represent quantitative aspects of the model. On the one hand, the specification formalism is general enough to include models like weighted modal transition systems and allows the system developers to employ even more complex label refinement than in previously studied theories. On the other hand, the formalism maintains the desirable properties required by any specification theory supporting compositional reasoning. In particular, we study modal and thorough refinement, determinization, parallel composition, conjunction, quotient, and logical characterization of label-structured modal transition systems.

3.4 Lynch-style timed I/O automata

Participants

- Dilsun Kaynar; Carnegie Mellon University
- Nancy Lynch; MIT
- Roberto Segala; University of Verona
- Frits Vaandrager; RU

Results The monograph [61], which appeared in Syntesis series of Morgan & Claypool, presents the *Timed Input/Output Automaton (TIOA)* modeling framework, a basic mathematical framework to support description and analysis of timed (computing) systems. Timed systems are systems in which desirable correctness or performance properties of the system depend on the timing of events, not just on the order of their occurrence. Timed systems are employed in a wide range of domains including communications, embedded systems, real-time operating systems, and automated control. Many applications involving timed systems have strong safety, reliability and predictability requirements, which makes it important to have methods for systematic design of systems and rigorous analysis of timing-dependent behavior.

The TIOA modeling framework presented in [61] evolved from the *Hybrid Input/Output Automaton (HIOA)* modeling framework for hybrid systems [69] by Lynch, Segala and Vaandrager. The HIOA framework, in turn, evolved from the I/O automata of [70, 71, 68, 49, 50], a fundamental modeling framework for (untimed) asynchronous systems. Our approach is based on the assumption that a timed system can be viewed as a special kind of a hybrid system where the continuous transformation is limited to internal system components that determine the timing of events. Therefore, we define a TIOA as a restricted HIOA where the only essential difference between an HIOA and a TIOA is that an HIOA may have *external variables* to model the continuous information flowing into and out of the system, in addition to state variables. A major consequence of this definition is that the communication between TIOAs is restricted to shared-action communication only. The TIOA model does not impose any further restrictions on the expressive power of the HIOA model.

The TIOA framework also supports description and analysis of timed distributed algorithms—distributed algorithms whose correctness and performance depend on the relative speeds of processors, on the accuracy of local clocks, or on communication delay bounds. Such algorithms arise, for example, in traditional and wireless communications, networks of mobile devices, and shared-memory multiprocessors. The need to prove rigorous theoretical results about timed distributed algorithms makes it important to have a suitable mathematical foundation.

An important feature of the TIOA framework is its support for decomposing timed system descriptions. In particular, the framework includes a notion of *external behavior* for a timed I/O automaton, which captures its discrete interactions with its environment. The framework also defines what it means for one TIOA to *implement* another, based on an inclusion relationship between their external behavior sets, and defines notions of *simulations*, which provide sufficient conditions for demonstrating implementation relationships. The framework includes a *composition* operation for TIOAs, which respects external behavior, and a notion of *receptiveness*, which implies that a TIOA does not block the passage of time.

The TIOA framework also defines the notion of a property and what it means for a property to be a safety or a liveness property. It includes results that capture common proof methods for showing that automata satisfy properties.

3.5 Three-valued abstraction of probabilistic systems

Participants

- Joost-Pieter Katoen, Daniel Klink; RWTH
- Martin Leucker; University of Lübeck
- Verena Wolf; SU

Results In [56], we study are discrete-time and continuous-time Markov chains (DTMCs and CTMCs, for short). DTMCs and CTMCs are a class of stochastic processes that are used to model and analyze random phenomena in application domains such as planning of production lines and safety-critical systems. A DTMC is a Kripke structure in which each transition is equipped with a discrete probability describing the likelihood of moving from one state to another in a single move. In addition, in a CTMC state residence times are governed by negative exponential distributions.

Abstraction is aimed at a model reduction by collapsing sets of concrete states to abstract states. Our abstraction technique is based on a partitioning of the concrete state space. Promising results in traditional model checking have been obtained for a three-valued semantics of temporal logic formulas, i.e., an interpretation in which a formula evaluates to either true, false, or indefinite. In this setting, abstraction is conservative for both positive and negative verification results. Only if the verification of the abstract model yields an indefinite answer ("don't know"), the validity in the concrete model is unknown. The abstraction techniques proposed in this paper follow this three-valued approach. For the discrete-time setting, we consider abstractions for the branching-time logic PCTL (Probabilistic Computation Tree Logic), whereas for the continuous-time case the logic CSL (Continuous Stochastic Logic) is regarded.

In classical model checking, three-valued abstraction yield abstract models (called modal transition systems) containing *may* and *must* transitions between aggregated states as over- and under-approximation, respectively, of the concrete transition relation. This concept can be lifted to DTMCs in a rather natural way by replacing transition probabilities by *intervals*. Lower and upper bounds of intervals now act as under- and over-approximation, respectively. In fact, the resulting abstract model is of interest on its own, since often only bounds on probabilities are known rather than precise values. States in abstract DTMCs are thus groups of concrete states and transitions are equipped with intervals. It is shown that concrete states are simulated—using Jonsson and Larsen's seminal notion of probabilistic simulation—by their abstract counterparts. Finally, a three-valued semantics of PCTL is provided which is proven to be appropriate for the abstraction considered in the sense that any affirmative or negative verification result on an abstract DTMC carries over to the concrete model. Our model-checking algorithm for checking

an abstract DTMC against a three-valued PCTL-formula is inspired by verification algorithms for MDPs.

A similar strategy is adopted for CTMCs. The main technical complication, however, is that besides transition probabilities, one has to determine the residence time of an abstract state that results from concrete states with possibly distinct residence times. We show that intervals of transition probabilities, intervals on residence times (or combinations thereof) are not satisfactory in terms of precision. Instead, we suggest to overcome this imprecision by using *uniform* CTMCs, i.e., CTMCs in which all states have equal residence times and use transition probability intervals. This is not a restriction, as any CTMC can be transformed into a weakly bisimilar uniform CTMC in linear time and weak bisimulation preserves the validity of CSL formulas except the next-step operator. The resulting abstraction is shown to preserve simulation: concrete states are simulated by their corresponding abstract ones. The next technical complication is that intervals offer schedulers infinitely many choices to select a transition probability. We show that extreme schedulers, i.e., schedulers that basically only consider lower- and upper bounds, suffice for reachability probabilities. Thus schedulers which have a finite number of choices suffice. Using a three-valued semantics of CSL, it is shown that the abstraction is indeed conservative for affirmative and negative verification results.

3.6 Time-bounded reachability in tree-structured QBDs by abstraction

Participants

- Daniel Klink, Joost-Pieter Katoen; RWTH
- Anne Remke, Boudewijn R. Haverkort; ESI/UT and ESI

Results Infinite-state Markov chains such as *tree-structured* quasi-birth death (QBD) processes have been applied to model single-server queues with a LIFO service discipline, to analyze random access algorithms, as well as priority queueing systems. Discrete-time tree-structured QBDs are equivalent to probabilistic pushdown automata and recursive Markov chains. The analysis of (tree-structured) QBDs mostly focuses on steady-state probabilities. Transient analysis has received scant attention; the only existing approach is approximate. Recently, direct techniques based on uniformization or variants thereof, have been proposed for reachability properties for general infinite-state CTMCs and for highly structured infinite-state CTMCs, such as normal (chain-like) QBDs and Jackson queueing networks. However, they all lead to an exponential blow-up when applied to tree-structured QBDs.

In [62, 63], we determine time-bounded reachability probabilities in tree-structured QBDs by applying CTMC abstraction. To that end, we consider two techniques, interval-based and MDP-based abstraction, and compare them. A major issue in state-based abstraction is to come up with an adequate, i.e., small though precise, partitioning of the state space. In fact, we identify various possibilities to partition the state space of a tree-structured QBD, relate them formally using simulation relations, and empirically investigate their influence on the accuracy of the obtained time-bounded reachability probabilities.

The partitioning methods range from simple schemes such as counter abstraction (group states with equal number of customers) to more advanced schemes in which the ordering of customers, and/or the service phase is respected. This yields tree-, chain-, and grid-like abstractions. We perform extensive experiments for phase-type service distributions of different orders and analyze the influence of parameter setting and partitioning scheme on the quality of the results and on the size of the resulting abstract state space. Our experiments show that grid-like schemes yield extremely precise approximations for rather coarse abstractions. It is shown that by adequate abstraction of the state space, transient probabilities in tree-based QBDs can be obtained by reducing a state space of over 10^{400} states by about 500,000 states while obtaining rather precise results. This clearlt shows the potential of this technique.

3.7 Compositional abstraction of stochastic systems

Participants

- Daniel Klink, Joost-Pieter Katoen, Martin R. Neuhäußer; RWTH

Results In [57], we propose a framework to perform aggressive abstraction of interactive Markov chains (IMCs) in a fully compositional manner. We consider state-based abstraction that allows to represent any (disjoint) group of concrete states by a single abstract state. This flexible abstraction mechanism generalizes bisimulation minimization (where only bisimilar states are grouped) and yields an over-approximation of the IMC under consideration. This abstraction is a natural mixture of abstraction of labeled transition systems by modal transition systems and abstraction of probabilities by intervals. Abstraction is shown to preserve simulation, that is to say, abstract models simulate concrete ones. Here, simulation is a simple combination of refinement of modal transition systems and probabilistic simulation. It is shown that abstraction yields lower bounds for minimal and upper bounds for maximal timed reachability probabilities.

Compositional aggregation is facilitated by the fact that simulation is a pre-congruence with respect to TCSP-like parallel composition and symmetric composition on our abstract model. Accordingly, components can be abstracted prior to composing them. As this abstraction is coarser than bisimulation, a significantly larger state-space reduction may be achieved and peak memory consumption is reduced. This becomes even more advantageous when components that differ only marginally are abstracted by the same abstract model. In this case, the symmetric composition of these abstract components may yield huge reductions compared to the parallel composition of the slightly differing concrete ones. A small example shows this effect, and shows that the obtained bounds for timed reachability probabilities are rather exact. Several abstraction techniques for (discrete) probabilistic models have been developed so far. However, compositional ones that go beyond bisimulation are rare. Notable exceptions are Segalas work on simulation preorders for probabilistic automata and language-level abstraction for PRISM.

3.8 Safety verification for probabilistic hybrid systems

Participants

- Lijun Zhang; DTU, Technical University of Denmark
- Zhikun She; LMIB and Beihang University, China
- Stefan Ratschan; Czech Academy of Sciences, Czech Republic
- Holger Hermanns, Ernst Moritz Hahn; SU

Context The interplay of random phenomena and continuous dynamics deserves increased attention especially in the context of wireless sensing and control applications. The analysis of properties of such systems thus needs to take into account probabilistic variations of systems with hybrid dynamics. In safety verification of classical hybrid systems one is interested in whether a certain set of unsafe system states can be reached from a set of initial states. In the probabilistic setting, we may ask instead whether the probability of reaching unsafe states is below some given threshold. Other important properties relate to the performance of systems, such as the average data transmission rate of a wireless multimedia application on the long run, or the expected maximal time a wireless sensing network needs to transmit a measurement to a base station.

Contribution In [86, 87] we have considered reachability properties for probabilistic hybrid systems. This model class extends classical hybrid systems by extending discrete jumps a probability distribution over the successor states. The system thus involves both probabilistic behaviour from these distributions as well as nondeterminism, because the time point when to execute an action and also which action to execute is still chosen nondeterministically. In addition, the continuous flow could also be nondeterministic. The underlying semantics of a probabilistic hybrid automaton is a Markov decision process with an uncountably large state space. Our target there was to verify that the maximal probability to reach a set of unsafe states is below a given threshold ε.

Because the state space is uncountable large, we could not directly apply standard means to compute extremal probabilities in Markov decision processes. To solve the problem under consideration, we computed a non-probabilistic version of a given probabilistic hybrid automaton, which is then fed into a usual solver for non-probabilistic hybrid systems, such as for instance PHAVER. In the abstraction obtained this way, we could re-attach probabilities to obtain a model which overapproximates the original semantics of the probabilistic hybrid automaton. Thus, if we could prove that the maximal reachability probability in the overapproximation is below ε, the same also holds for the semantics. To demonstrate the practical applicability of the approach, we applied our prototype tool PROHVER on several small case studies.

This work is developed jointly with the German special research initiative AVACS subproject H4. It is published in *CAV 2010* [86]. An extended version will appear in the *European Journal of Control* [87].

Perspective We are planning to extend the approach in several directions, and indeed have done so recently [44]. PROHVER needs to become robust and user friendly and to allow more

hybrid solvers to be supported to compute the abstraction. We are also planning to extend the class of properties to be handled.

4 Counterexamples and CEGAR

4.1 Counterexample generation in probabilistic model checking

Participants

- Tingting Han, Joost-Pieter Katoen, Berteun Damman; RWTH

Results A major strength of model checking is the possibility to generate counterexamples in case a property is violated. They are of utmost importance in model checking: first, and for all, they provide diagnostic feedback even in cases where only a fragment of the entire model can be searched. They also constitute the key to successful abstraction-refinement techniques, and are at the core of obtaining feasible schedules in e.g., timed model checking. As a result, advanced counterexample generation and analysis techniques have intensively been investigated.

The shape of a counterexample depends on the checked formula and the temporal logic. For logics such as LTL, typically finite paths through the model suffice. The violation of linear-time safety properties is indicated by finite paths that end in a "bad" state. Liveness properties, instead, require infinite paths ending in a cyclic behavior indicating that something "good" will never happen. LTL model checkers usually incorporate breadth-first search algorithms to generate *shortest* counterexamples, i.e., paths of minimal length. For branching-time logics such as CTL, paths may act as counterexamples for a subclass of universally quantified formulae, i.e., those in ACTL and LTL. To cover a broader spectrum of formulae, though, more advanced structures such as trees of paths, proof-like counterexamples (for ACTL $\setminus$ LTL) or annotated paths (for existential CTL) are used.

In [45], we consider the generation of counterexamples in probabilistic model checking. The crux of probabilistic model checking is to appropriately combine techniques from numerical mathematics and operations research with standard reachability analysis. In this way, properties such as "the (maximal) probability to reach a set of goal states by avoiding certain states is at most 0.6" can be automatically checked up to a user-defined precision. Markovian models comprising millions of states can be checked rather fast by dedicated tools such as PRISM and MRMC, as well as extensions to existing tools such as GreatSPN, SPIN and PEPA Workbench.

In probabilistic model checking, however, counterexample generation is almost not developed; notable exception is the recent heuristic search algorithm for Markov chains that works under the assumption that the model is unknown. Instead, we consider a setting in which it has already been established that a certain state refutes a given property. This paper considers algorithms, complexity results, and experimental results for the generation of counterexamples in probabilistic model checking. The considered setting is probabilistic CTL for DTMCs. In this setting, typically there is no single path but rather a *set* of paths that indicates why a given property is refuted. We first concentrate on properties of the form $\mathcal{P}_{\leq p}(\Phi\ \mathrm{U}^{\leq h}\ \Psi)$ where Φ and

Ψ characterize sets of states, p is a probability, and h a (possibly infinite) bound on the maximal allowed number of steps before reaching a goal (i.e., a Ψ-) state. In case state s refutes this formula, the probability of all paths in s satisfying $\Phi\ \mathrm{U}^{\leq h}\ \Psi$ exceeds p. We consider two problems that are aimed to provide useful diagnostic feedback for this violation: generating strongest evidences and smallest, most indicative counterexamples.

Similar to the notion of shortest counterexample in LTL model checking, we consider trees of *smallest size* that exceed the probability bound p. Additionally, such trees, of size k, say, are required to *maximally* exceed the lower bound, i.e., no subtrees should exist of size at most k that exceed p more. The problem of generating such *smallest, most indicative counterexamples* can be cast as a k shortest paths problem. For unbounded-until formulae (i.e., $h{=}\infty$), the generation of such smallest counterexamples can be carried out in pseudo-polynomial time by adopting k shortest paths algorithms that compute k on the fly. For bounded until-formulae, we propose an adaptation of the recursive enumeration algorithm (REA) of Jiménez and Marzal. The time complexity of this adapted algorithm is $\mathcal{O}(hm{+}hk\log(\frac{m}{n}))$, where n is the number of states in the DTMC.

This approach is applicable to probability thresholds with lower bounds, as well as to the logic LTL. It is applicable to various other models such as Markov reward models and Markov decision processes (MDPs), once a scheduler for a violating an until-formula is obtained. It also provides the basis for counterexample generation techniques for time-bounded reachability in CTMCs, CEGAR techniques for MDPs, and counterexamples for the logic cpCTL.

Once we have established the theoretical underpinnings, we report on experiments that apply our counterexample generation algorithms to example DTMCs. Using the synchronous leader election protocol by Itai and Rodeh, we show that the size of counterexamples may be double exponential in terms of the input parameters of the protocol (like number of processes and rounds). To achieve a more succinct representation we propose to use regular expressions. The advantage of regular expressions is that they are commonly known, are easy to understand, and may be very compact. The idea is to represent a DTMC by a deterministic finite-state automaton (DFA) and obtain regular expressions by applying successive state elimination. The computation of the probability of a regular expression is performed using the approach advocated by Daws for parametric model checking of DTMCs. This boils down to a recursive evaluation which is guaranteed to be exact (i.e., no rounding errors), provided the transition probabilities are rational.

4.2 Abstraction refinement for infinite probabilistic models

Participants

- Ernst Moritz Hahn, Holger Hermanns, Björn Wachter; SU
- Lijun Zhang; DTU, Technical University of Denmark

Context Network protocols are subject to random phenomena like unreliable communication and employ randomization as a strategy for collision avoidance. Further, they are often distributed and thus inherently concurrent. To account for both randomness and concurrency,

Markov decision processes (MDPs) are used as a semantic foundation as they feature both non-deterministic and probabilistic choice. Typically one is interested in computing (maximal or minimal) reachability probabilities, e.g., of delivering three messages after ten transmission attempts (under best-case and worst-case assumptions concerning the environment). Probabilistic reachability is expressible in terms of least fixed points of a system of recursive equations where the unknowns correspond to the probability of an individual state. For finite MDPs, probabilistic reachability can be reduced to linear programming or solved approximately by value iteration. However, the state explosion problem is a major problem, in fact, even more than in a qualitative setting as computing probabilities entails expensive numerical computations, where symbolic techniques are not advantageous.

Contribution We have developed PASS, a tool that analyzes concurrent probabilistic programs, which map to potentially infinite Markov decision processes. PASS is based on predicate abstraction and abstraction refinement. As such, it implements Quasimodo progress described in Deliverable D2.1 (Section 4.4) and Deliverable D2.3 (Section 4.1). PASS scales to programs far beyond the reach of numerical methods which operate on the full state space of the model. The computational engines we use are SMT solvers to compute finite abstractions, numerical methods to compute probabilities and interpolation as part of abstraction refinement. PASS has been successfully applied to network protocols and serves as a test platform for different refinement methods. This work has been published in [43]. PASS consists of approximately 18.000 lines of C++ code, and has been tested on a large number of case studies.

Perspective There are several directions in which we are planning to extend PASS. Our tool can in principal handle variable-annotated, probabilistic timed automata, by using the digital clock semantics [65]. The abstraction of the non-clock variable turns out to work as well as for non-timed models. However, because of the clock variables, the refinement process includes an excessive number of extra predicates. We are going to attack this problem by separating the clock variables from the other variables, thus avoiding the problem and enabling us to apply our technique on models with complex timing behaviour.

References

[1] C. Baier, F. Ciesinski, and M. Größer. PROBMELA: a modeling language for communicating probabilistic processes. In *Proc. of the 2nd ACM/IEEE Int. Conf. on Formal Methods and Models for Co-Design (MEMOCODE)*, pages 57–66. IEEE, 2004.

[2] C. Baier, P.R. D'Argenio, and M. Größer. Partial order reduction for probabilistic branching time. In *Proc. of the 3rd Workshop on Quantitative Aspects of Programming Languages (QAPL)*, volume 153(2) of *ENTCS*, pages 97–116, 2006.

[3] C. Baier, M. Größer, and F. Ciesinski. Partial order reduction for probabilistic systems. In *Proc. of the 1st International Conference on Quantitative Evaluation of Systems (QEST)*, pages 230–239. IEEE Computer Society, 2004.

[4] Sebastian S. Bauer, Line Juhl, Kim G. Larsen, Axel Legay, and Jiri Srba. Extending modal transition systems with structured labels. 2011. Under submission.

[5] Gerd Behrmann, Agnès Cougnard, Alexandre David, Emmanuel Fleury, Kim Guldstrand Larsen, and Didier Lime. Uppaal-tiga: Time for playing games! In Werner Damm and Holger Hermanns, editors, *Computer Aided Verification, 19th International Conference, CAV 2007, Berlin, Germany, July 3-7, 2007, Proceedings*, volume 4590 of *Lecture Notes in Computer Science*, pages 121–125. Springer, 2007.

[6] M. Bezem and J. F. Groote. Invariants in process algebra with data. In *Proc. of the 5th Int. Conf. on Concurrency Theory (CONCUR)*, volume 836 of *LNCS*, pages 401–416. Springer, 1994.

[7] S. C. C. Blom. Partial τ-confluence for efficient state space generation. Technical Report SEN-R0123, CWI, 2001.

[8] S. C. C. Blom, B. Lisser, J. C. van de Pol, and M. Weber. A database approach to distributed state-space generation. *Journal of Logic and Computation*, 21:45–62, 2011.

[9] S. C. C. Blom and J. C. van de Pol. State space reduction by proving confluence. In *Proc. of the 14th Int. Conf. on Computer Aided Verification (CAV)*, volume 2404 of *LNCS*, pages 596–609. Springer, 2002.

[10] S. C. C. Blom and J. C. van de Pol. Symbolic reachability for process algebras with recursive data types. In *Proc. of the 5th Int. Colloquium on Theoretical Aspects of Computing (ICTAC)*, volume 5160 of *LNCS*, pages 81–95. Springer, 2008.

[11] H. C. Bohnenkamp, P. R. D'Argenio, H. Hermanns, and J.-P. Katoen. MODEST: A compositional modeling formalism for hard and softly timed systems. *IEEE Transactions of Software Engineering*, 32(10):812–830, 2006.

[12] D.J.B. Bosscher and A. Ponse. Translating a process algebra with symbolic data values to linear format. In U.H. Engberg, K.G. Larsen, and A. Skou, editors, *Proceedings of the Workshop on Tools and Algorithms for the Construction and Analysis of Systems,* Aarhus, Denmark, volume NS-95-2 of *BRICS Notes Series*, pages 119–130. Department of Computer Science, University of Aarhus, May 1995.

[13] Benoit Caillaud, Benoit Delahaye, Kim G. Larsen, Mikkel Larsen Pedersen, and Andrzej Wasowski. Compositional design methodology with constraint markov chains. under submission.

[14] Bernoit Caillaud, Bernoit Caillaud, Bernoit Delahaye, Kim G. Larsen, Axel Legay, Mikkel Larsen Pedersen, and Andrzej Wasowski. Constraint markov chains. *Theoretical Computer Science*, 2011. To appear.

[15] F. Cassez, A. David, E. Fleury, K.G. Larsen, and D. Lime. Efficient on-the-fly algorithms for the analysis of timed games. In Martín Abadi and Luca de Alfaro, editors, *CONCUR 2005 - Concurrency Theory, 16th International Conference, CONCUR 2005, San Francisco, CA, USA, August 23-26, 2005, Proceedings*, volume 3653 of *Lecture Notes in Computer Science*, pages 66–80. Springer, 2005.

[16] Rohit Chadha and Mahesh Viswanathan. A counterexample-guided abstraction-refinement framework for markov decision processes. *ACM Trans. Comput. Log.*, 12(1):1, 2010.

[17] E.M. Clarke, A. Fehnker, Z. Han, B.H. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald. Abstraction and counterexample-guided refinement in model checking of hybrid systems. *Int. J. Found. Comput. Sci.*, 14(4):583–604, 2003.

[18] E.M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In E.A. Emerson and A.P. Sistla, editors, *Computer Aided Verification, 12th International Conference, CAV 2000, Chicago, IL, USA, July 15-19, 2000, Proceedings*, volume 1855 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2000.

[19] E.M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement for symbolic model checking. *J. ACM*, 50(5):752–794, 2003.

[20] P. R. D'Argenio, B. Jeannet, H. E. Jensen, and K. G. Larsen. Reachability analysis of probabilistic systems by successive refinements. In *Proc. of the Joint Int. Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification (PAPM-PROBMIV)*, volume 2165 of *LNCS*, pages 39–56. Springer, 2001.

[21] P.R. D'Argenio and P. Niebert. Partial order reduction on concurrent probabilistic programs. In *Proc. of the 1st International Conference on Quantitative Evaluation of Systems (QEST)*, pages 240–249. IEEE Computer Society, 2004.

[22] Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. ECDAR: An environment for compositional design and analysis of real time systems. In *Proceedings of Automated Technology for Verification and Analysis*, volume 6252 of *LNCS*, pages 365–370. Springer, 2010.

[23] Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. Timed i/o automata: A complete specification theory for real-time systems. In *Proceedings of Hybrid Systems: Computation and Control*. ACM, 2010.

[24] Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, Andrzej Wasowski, Timothy Bourke, and Didier Lime. New results on timed specifications. 2011. To appear in Post-Proceedings of WADT 2011.

[25] L. de Alfaro and P. Roy. Magnifying-lens abstraction for Markov decision processes. In *Proc. of the 19th Int. Conf. on Computer Aided Verification (CAV)*, volume 4590 of *LNCS*, pages 325–338. Springer, 2007.

[26] Benoit Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel Larsen Pedersen, F. Sher, and Andrzej Wasowski. Abstract probabilistic automata. 2011. To appear in Proceedings of VMCAI 2011.

[27] Benoit Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel Larsen Pedersen, F. Sher, and Andrzej Wasowski. New results on abstract probabilistic automata. 2011. To appear in Proceedings of ACDC 2011.

[28] Benoit Delahaye, Kim G. Larsen, Axel Legay, Mikkel Larsen Pedersen, and Andrzej Wasowski. Apac: a tool for reasoning about abstract probabilistic automata. 2011. To appear in Proceedings of QEST 2011.

[29] Benoit Delahaye, Kim G. Larsen, Axel Legay, Mikkel Larsen Pedersen, and Andrzej Wasowski. Decision problems for interval markov chains. 2011. To appear in Proceedings of LATA 2011.

[30] Christian Eisentraut, Holger Hermanns, and Lijun Zhang. Concurrency and composition in a stochastic world. In Paul Gastin and Franois Laroussinie, editors, *CONCUR 2010 - Concurrency Theory*, volume 6269 of *Lecture Notes in Computer Science*, pages 21–39. Springer Berlin / Heidelberg, 2010.

[31] Christian Eisentraut, Holger Hermanns, and Lijun Zhang. On probabilistic automata in continuous time. In *Logic in Computer Science, Symposium on*, pages 342–351, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[32] M. V. Espada and J. C. van de Pol. An abstract interpretation toolkit for μCRL. *Formal Methods in System Design*, 30(3):249–273, 2007.

[33] W. Fokkink, J. Pang, and J. C. van de Pol. Cones and foci: A mechanical framework for protocol verification. *Formal Methods in System Design*, 29(1):1–31, 2006.

[34] H. Garavel and M. Sighireanu. Towards a second generation of formal description techniques - rationale for the design of E-LOTOS. In *Proc. of the 3rd Int. Workshop on Formal Methods for Industrial Critical Systems (FMICS)*, pages 198–230. CWI, 1998.

[35] S. Giro, P.R. D'Argenio, and L. María Ferrer Fioriti. Partial order reduction for probabilistic systems: A revision for distributed schedulers. In *Proc. of the 20th International Conference on Concurrency Theory (CONCUR)*, volume 5710 of *LNCS*, pages 338–353. Springer, 2009.

[36] J. F. Groote and B. Lisser. Computer assisted manipulation of algebraic process specifications. Technical Report SEN-R0117, CWI, 2001.

[37] J. F. Groote and R. Mateescu. Verification of temporal properties of processes in a setting with data. In *Proc. of the 7th Int. Conf. on Algebraic Methodology and Software Technology (AMAST)*, volume 1548 of *LNCS*, pages 74–90. Springer, 1998.

[38] J. F. Groote and A. Ponse. The syntax and semantics of μCRL. In *Proc. of Algebra of Communicating Processes*, Workshops in Computing, pages 26–62. Springer, 1995.

[39] J. F. Groote and J. Springintveld. Focus points and convergent process operators: a proof strategy for protocol verification. *Journal of Logic and Algebraic Programming*, 49(1-2):31–60, 2001.

[40] J. F. Groote and T. A. C. Willemse. Model-checking processes with data. *Science of Computer Programming*, 56(3):251–273, 2005.

[41] J.F. Groote and M.P.A. Sellink. Confluence for process verification. *Theoretical Computer Science*, 170(1-2):47–81, 1996.

[42] M. Größer. *Reduction Methods for Probabilistic Model Checking*. PhD thesis, Technische Universität Dresden, 2008.

[43] Ernst Moritz Hahn, Holger Hermanns, Björn Wachter, and Lijun Zhang. Pass: Abstraction refinement for infinite probabilistic models. In Javier Esparza and Rupak Majumdar, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 16th International Conference, TACAS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings*, volume 6015 of *Lecture Notes in Computer Science*, pages 353–357. Springer, 2010.

[44] Ernst Moritz Hahn, Gethin Norman, David Parker, Björn Wachter, and Lijun Zhang. Game-based abstraction and controller synthesis for probabilistic hybrid systems. In *QEST*, 2011.

[45] T. Han, J.P. Katoen, and B. Damman. Counterexample generation in probabilistic model checking. *IEEE transactions on software engineering*, pages 241–257, 2009.

[46] T. A. Henzinger, M. Mateescu, and V. Wolf. Sliding window abstraction for infinite Markov chains. In *Proc. of the 21st Int. Conf. on Computer Aided Verification (CAV)*, volume 5643 of *LNCS*, pages 337–352. Springer, 2009.

[47] H. Hermanns, B. Wachter, and L. Zhang. Probabilistic CEGAR. In *Proc. of the 20th Int. Conf. on Computer Aided Verification (CAV)*, volume 5123 of *LNCS*, pages 162–175. Springer, 2008.

[48] A. Hinton, M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In *Proc. of the 12th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 3920 of *LNCS*, pages 441–444. Springer, 2006.

[49] B. Jonsson. Modular verification of asynchronous networks. In PODC'87 [77], pages 152–166.

[50] B. Jonsson. Compositional specification and verification of distributed systems. *ACM Transactions on Programming Languages and Systems*, 16(2):259–303, March 1994.

[51] B. Jonsson and K.G. Larsen. Specification and refinement of probabilistic processes. In *Proceedings* 6^{th} *Annual Symposium on Logic in Computer Science,* Amsterdam, pages 266–277. IEEE Press, 1991.

[52] J.-P. Katoen, D. Klink, M. Leucker, and V. Wolf. Three-valued abstraction for continuous-time Markov chains. In *Proc. of the 19th Int. Conf. on Computer Aided Verification (CAV)*, volume 4590 of *LNCS*, pages 311–324. Springer, 2007.

[53] J.-P. Katoen, J. C. van de Pol, M. I. A. Stoelinga, and M. Timmer. A linear process-algebraic format for probabilistic systems with data. In *Proc. of the 10th Int. Conf. on Application of Concurrency to System Design (ACSD)*, pages 213–222. IEEE, 2010.

[54] J.-P. Katoen, J. C. van de Pol, M. I. A. Stoelinga, and M. Timmer. A linear process algebraic format for probabilistic systems with data (extended version). Technical report, TR-CTIT-10-11, CTIT, University of Twente, 2010.

[55] J.-P. Katoen, J.C. van de Pol, M.I.A. Stoelinga, and M. Timmer. A linear process-algebraic format with data for probabilistic automata. *Theoretical Computer Science*, 2011 (to appear).

[56] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Three-valued abstraction for probabilistic systems. *Journal on Logic and Algebraic Programming*, pages 1–55, 2011.

[57] Joost-Pieter Katoen, Daniel Klink, and Martin Neuhuer. Compositional abstraction for stochastic systems. In Joël Ouaknine and Frits Vaandrager, editors, *Formal Modeling and Analysis of Timed Systems*, volume 5813 of *Lecture Notes in Computer Science*, pages 195–211. Springer Berlin / Heidelberg, 2009.

[58] M. Kattenbelt, M. Z. Kwiatkowska, G. Norman, and D. Parker. Abstraction refinement for probabilistic software. In *Proc. of the 19th Int. Conf. on Verification, Model Checking, and Abstract Interpretation (VMCAI)*, volume 5403 of *LNCS*, pages 182–197. Springer, 2009.

[59] M. Kattenbelt, M. Z. Kwiatkowska, G. Norman, and D. Parker. A game-based abstraction-refinement framework for Markov decision processes. *Formal Methods in System Design*, 36(3):246–280, 2010.

[60] Mark Kattenbelt, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. A game-based abstraction-refinement framework for markov decision processes. *Formal Methods in System Design*, 36(3):246–280, 2010.

[61] D.K. Kaynar, N.A. Lynch, R. Segala, and F.W. Vaandrager. *The Theory of Timed I/O Automata (second edition)*. Morgan & Claypool Publishers, 2010. Synthesis Lecture on Distributed Computing Theory, 137pp.

[62] Daniel Klink, Anne Remke, Boudewijn R. Haverkort, and Joost-Pieter Katoen. Time-bounded reachability in tree-structured qbds by abstraction. In *Quantitative Evaluation of Systems (QEST)*, pages 133–142. IEEE CS Press, 2009.

[63] Daniel Klink, Anne Remke, Boudewijn R. Haverkort, and Joost-Pieter Katoen. Time-bounded reachability in tree-structured qbds by abstraction. *Performance Evaluation*, 68(2):105–125, 2011.

[64] R.P. Kurshan. *Computer-Aided Verification of Coordinating Processes*. Princeton University Press, 1994.

[65] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. *Theoretical Computer Science*, 282:101–150, 2002.

[66] M. Z. Kwiatkowska, G. Norman, and D. Parker. Game-based abstraction for Markov decision processes. In *Proc. of the 3rd Int. Conf. on Quantitative Evaluation of Systems (QEST)*, pages 157–166. IEEE, 2006.

[67] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991.

[68] N.A. Lynch. Modelling and verification of automated transit systems, using timed automata, invariants and simulations. In R. Alur, T.A. Henzinger, and E.D. Sontag, editors, *Hybrid Systems III*, volume 1066 of *Lecture Notes in Computer Science*, pages 449–463. Springer-Verlag, 1996.

[69] N.A. Lynch, R. Segala, and F.W. Vaandrager. Hybrid I/O automata. *Information and Computation*, 185(1):105–157, 2003.

[70] N.A. Lynch and M.R. Tuttle. Hierarchical correctness proofs for distributed algorithms. In PODC'87 [77], pages 137–151. A full version is available as MIT Technical Report MIT/LCS/TR-387.

[71] N.A. Lynch and M.R. Tuttle. An introduction to input/output automata. *CWI Quarterly*, 2(3):219–246, September 1989.

[72] Radu Mardare, Luca Cardelli, and Kim G. Larsen. Modular markovian logic. 2011. To appear in Proceedings of ICALP 2011.

[73] C. Morgan and A. McIver. pGCL: formal reasoning for random algorithms. *South African Computer Journal*, 22:1427, 1999.

[74] R. De Nicola and F.W. Vaandrager. Action versus state based logics for transition systems. In *Semantics of Systems of Concurrent Processes*, volume 469 of *LNCS*, pages 407–419. Springer, 1990.

[75] G.J. Pace, F. Lang, and R. Mateescu. Calculating τ-confluence compositionally. In *Proc. of the 15th International Conference on Computer Aided Verification (CAV)*, volume 2725 of *LNCS*, pages 446–459. Springer, 2003.

[76] D. Peled. All from one, one for all: on model checking using representatives. In *Proc. of the 5th International Conference on Computer Aided Verification (CAV)*, volume 697 of *LNCS*, pages 409–423. Springer, 1993.

[77] *Proceedings of the* 6^{th} *Annual ACM Symposium on Principles of Distributed Computing*, August 1987.

[78] Jean-Baptiste Raclet, Eric Badouel, Albert Benveniste, Benoît Caillaud, Axel Legay, and Roberto Passerone. Modal interfaces: unifying interface automata and modal specifications. In Samarjit Chakraborty and Nicolas Halbwachs, editors, *Proceedings of the 9th ACM & IEEE International conference on Embedded software, EMSOFT 2009, Grenoble, France, October 12-16, 2009*, pages 87–96. ACM, 2009.

[79] S. Roolvink, A.K.I. Remke, and M.I.A. Stoelinga. Dependability and survivability evaluation of a water distribution process with arcade. In *Proceedings of the Eighth International Workshop on Performability Modeling of Computer and Communication Systems (PMCCS'09)*, 2009.

[80] R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 1995. Available as Technical Report MIT/LCS/TR-676.

[81] M. Timmer, M. I. A. Stoelinga, and J. C. van de Pol. Confluence reduction for probabilistic systems (extended version). Technical Report 1011.2314, ArXiv e-prints, 2010.

[82] M. Timmer, M. I. A. Stoelinga, and J. C. van de Pol. Confluence reduction for probabilistic systems. In *Proc. of the 17th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 6605 of *LNCS*, pages 311–325. Springer, 2011.

[83] Y. S. Usenko. *Linearization in μCRL*. PhD thesis, Eindhoven University of Technology, 2002.

[84] J. C. van de Pol and M. Timmer. State space reduction of linear processes using control flow reconstruction. In *Proc. of the 7th Int. Symp. on Automated Technology for Verification and Analysis (ATVA)*, volume 5799 of *LNCS*, pages 54–68. Springer, 2009.

[85] P. C. W. van den Brand, M. A. Reniers, and P. J. L. Cuijpers. Linearization of hybrid processes. *Journal of Logic and Algebraic Programming*, 68(1-2):54–104, 2006.

[86] Lijun Zhang, Zhikun She, Stefan Ratschan, Holger Hermanns, and Ernst Hahn. Safety verification for probabilistic hybrid systems. In Tayssir Touili, Byron Cook, and Paul Jackson, editors, *Computer Aided Verification*, volume 6174 of *Lecture Notes in Computer Science (LNCS)*, pages 196–211. Springer Berlin / Heidelberg, 2010.

[87] Lijun Zhang, Zhikun She, Stefan Ratschan, Holger Hermanns, and Ernst Moritz Hahn. Safety verification for probabilistic hybrid systems. *European Journal of Control*, 2011.